Creating a Design Document

Throughout this course, you will be completing several programming projects in as a team. To facilitate making progress and moving towards the creation and testing of your projects, you are required to submit a *Design Document* two weeks before the programming part of the assignment is due. This is a team document, and everyone on the team will receive the same grade.

Note that the Design Document makes up 30% of the project grade. Each of the following sections lists their corresponding point values as a grading rubric.

Your design document is intended to help you, not be a burden. Therefore, the document can be somewhat informal in its presentation. Even so, we expect you to take pride in what you submit and to be sure all of the following elements are included in your document. Given the informality, you may use whatever format you want, with the following exceptions:

1. Use $8\frac{1}{2} \times 11$ page size
2. Use 12-point font, preferrably TImes New Roman or similar.
3. Single space all text, and make sure your text is actual prose (not just bullets).
4. Separate each section with an appropriate section header in **Bold** font.
5. Do not "screen capture" any graphics. Use only high-quality graphics.
6. Make sure your figures or tables have captions. Then refer to the figure/table number in the text and explain the figure/table.
7. If you have any math, use built-in math editing capabilities.

1 Header Information (5%)

Think of the header as being what would go on the title page, but no separate title page is required. This includes the following:

1. Your team number. (If you want to include a clever name for your team, that is ok too, but the number is required.)
2. The full names (first, last) of all members of the team. (Email addresses may be provided but are not required since we can reach you through Brightspace.)
3. Identification of the project (i.e., Project 1, Project 2, Project 3, Project 4)
4. The date the document is submitted.

2 System Requirements (15%)

Record what you perceive to be the requirements of the project. Much of this section can come straight from the assignment itself. You may use the language from the assignment verbatim, but you cannot stop there. In addition to reporting the requirements, we also want you to provide how you understand each of the requirements as well as provide a high-level description of how you intend to satisfy each requirement.

3 System Architecture (20%)

At this point in the process, you should have a sense of how you want to approach developing your system, and this section is intended to document the architecture of that system. For this requirement, you have a choice between providing a UML class diagram (if you are oriented towards utilizing an object-oriented approach) or a functional block diagram (which could be utilized in either an object-oriented or non-object-oriented design). Whichever approach you choose, your architecture description must include the following:

1. A graphical layout of the architecture (i.e., the diagram requested above).
2. A textual, prose-based description of each of the major elements within your architecture (what is it, what does it do, what are the inputs, what are the outputs, what are the major methods or procedures, what requirement(s) are being satisfied by each part).

4 System Flow (20%)

Once you have the high-level architecture designed, the next step is to walk through how the system will be used to solve a target problem. Note that we are *not* asking you to walk through the actual solution to one of the problems. Rather, we want you to trace through the design, explaining how each of the components will be used in solving a problem. This is related to Section 3 in that it expands on each of the components more from a process flow point of view. It will also be helpful in responding to Section 5 since it identifies the key functions to be tested. You can do this descriptively or via something like a data-flow diagram (with associated explanations). For the more UML-oriented person, a high-level sequence diagram (with explanations) could also be quite effective. In the end, the choice is yours.

5 Test Strategy (20%)

This relates back to Section 2. Here, the intent is for you to lay out your approach to testing whether or not each of the requirements have been satisfied. Note that this is different from unit testing or interface testing, since you haven't gotten to the point where that level of the detail in the design exists yet. Instead, lay out a general procedure that, ultimately, will be reported on in your final project report, focused on each requirement from a functional point of view. Note that you can draw minimally from the elements you need to include in the video, but the intent is to address all of the major functional test elements of your project.

6 Task Assignments and Schedule (10%)

One of the challenges in any team-based project comes down to workload distribution. Ultimately, you are expected to share the workload equally on all of the assignments. In this section, you will specify who is expected to do what. You will identify the main tasks and assign those tasks to members of the team. Be sure to explain what each task is supposed to accomplish, especially with respect to the requirements discussed in Section 2.

Note that in your final report, you will also explain to what extent each member of the team accomplished the tasks that were assigned, so think very carefully about this.

You should also lay out an expected schedule for completing the tasks. This is usually best done through something like a Gantt Chart, but it's your choice how you relay this information.

Finally, describe what communication processes will be used to ensure the team continues to work together and make progress.

7 Other (10%)

Keep the following things in mind as you finish up your documents.

- You are *not permitted* to use AI-based tools of any kind, such as ChatGPT or Co-Pilot. All parts of all projects must be entirely the work of your team.
- You are *not permitted* to use algorithm-based libraries of any kind that implement the algorithms assigned. You must implement these algorithms entirely from scratch. When in doubt, ask.
- If you use any external resources, including ones you're not supposed to use, you must cite them and provided a "Reference" section. The format doesn't matter as long as it is complete. Failure to do so is a form of academic misconduct.
- Please proofread your document(s) carefully before turning them in. Sloppy, poorly organized, non-grammatical, error-ridden documents reflect poorly on the team. This is the major gradable element here.

If you have any questions, reach out to the TA or the instructor for clarification.